

## **REMARKS**

### **Request for Withdrawal of Final Action**

Applicants respectfully request reconsideration of the finality of the rejections in the Final Action dated January 2, 2008. Applicants submit that all subject matter introduced into the claims in the Amendment dated October 15, 2007 was fully supported by the specification as filed, and that the Examiner's rejections under 35 U.S.C. 112 first paragraph were improper. Therefore, Applicants submit that the Final Action dated January 2, 2008 was premature, and request that the previously submitted amendments be entered and considered.

### **Rejections under 35 U.S.C. 112 first paragraph**

The Examiner rejected claims 1, 3, 4, 5 and 7 - 15 for containing subject matter which was not described in the specification in such a way as to reasonably convey to one skilled in the art that the inventors had possession of the claimed invention.

Applicants respectfully disagree. Applicants submit that all the amendments are fully supported by the specification as originally filed, which describes the subject matter in such a way as to reasonably convey to one skilled in the art that the inventors were in full possession of the claimed invention at the date of filing.

The Examiner made specific rejections to claims 1, 4, 9, 10 14 and 15, which are addressed in order. Claims 1, 4, 9, 10 14 and 15 are reproduced below with underlining to indicate the amendments made in the Amendment dated October 15, 2007.

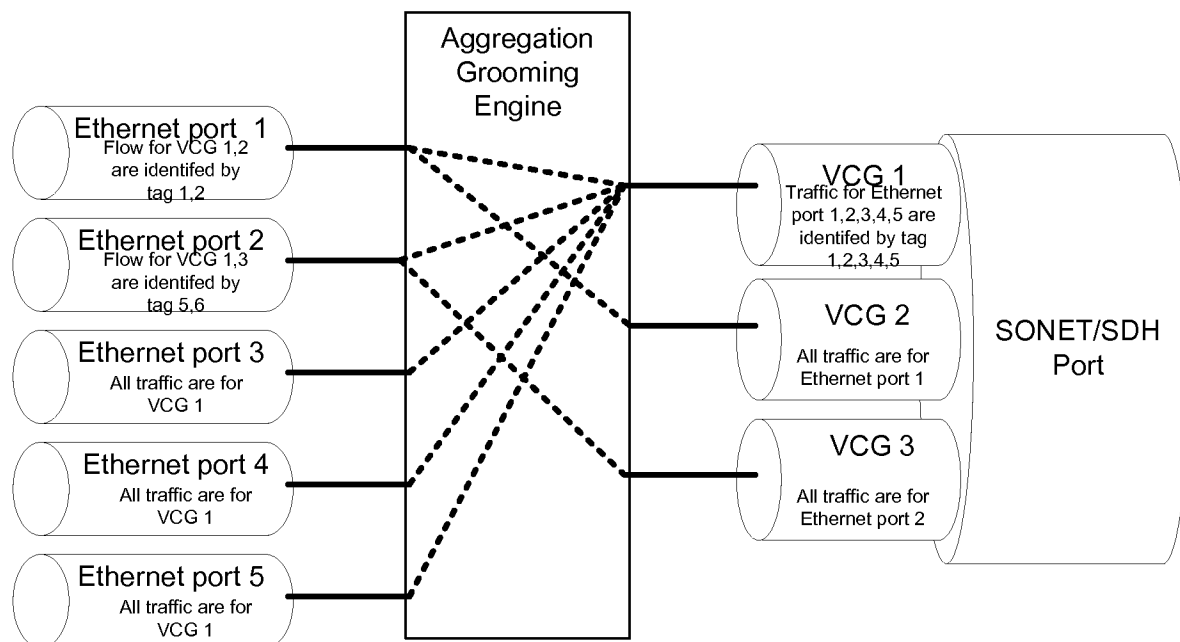
### ***Claim 1***

Amended claim 1 reads:

1. A method of packet grooming and aggregation within an Ethernet over SONET/SDH system (EOS system), said method comprising:  
receiving data packets each tagged according to an encapsulation scheme and including a port or channel ID;  
multiplexing a number of data streams according to respective tags,  
port or channel IDs of said data packets; and

mapping each said data stream directly to a physical transport interface by tag modification independent of any Layer 2 bridging or Layer 3 routing protocol.

Support for the amendments to claim 1 can be found in at least in paras. [0059] - [0064], and Fig. 6, of the specification as filed. For the Examiner's convenience, the relevant paragraphs and drawing are reproduced below:



**FIGURE 6**

**[0059]** The Aggregation/Grooming Engine (AGE) provides the mapping of client flows with QoS assurance between the Ethernet ports and the SONET/SDH Virtual Concatenation Groups (VCG). As shown in the Aggregation/Grooming Data Flow of FIGURE 6, at the ingress direction the AGE takes traffic aggregate from the Ethernet ports consisting of one or multiple client flows, grooming each flow according to flow tag of certain format, and aggregate these flows into the specified SONET/SDH VCGs according to the QoS configuration. On the egress direction, the traffic aggregates from the VCGs are groomed first into client flows, and then aggregate into the Ethernet ports.

**[0060]** With further reference to FIGURE 6, the traffic aggregate from/to the VCG and Ethernet ports are represented by solid lines, the client

flows are represented by dotted lines. For instance, the traffic from Ethernet port 1 contains two multiplexed client packet flows identified by tag value 1 and 2 respectively. The flow 1 is to be mapped into SONET/SDH VCG 1 after aggregation with many client flows from Ethernet port 2, 3, 4, and 5. The client flow 2 from Ethernet port 1 is mapped into SONET/SDH VCG 2 that is dedicated to carry this single flow. As another example, at the egress direction, the traffic from VCG 1 consists of 5 multiplexed packet flows identified by tag 1, 2, 3, 4 and 5. These flows are groomed and buffered separately and then sent to the appropriate Ethernet ports after merging with client flows that share the same Ethernet ports. For Ethernet port 1, the client flow 1 from VCG 1 and the flow from VCG 2 are merged (aggregated).

**[0061]** One embodiment of the Aggregation/Grooming Engine structure (based on unidirectional Lookup Engines) is shown in FIGURE 7. The Aggregation/Grooming Engine is divided into two symmetrical subsystems: Ingress AGE and Egress AGE. The Ingress AGE has the same structure as the Egress AGE, but may differ slightly in regard to the tag and frame format related to the Header Unit and the Tag editor. It should be noted that the Ingress AGE and Egress AGE may either use two separate designs, or use variations of the same design that supports the frame and tag format of both the ingress and egress.

**[0062]** An alternative embodiment of the AGE based on a Bi-directional Lookup Engine is shown in FIGURE 8. The only difference with the first AGE embodiment is in the Lookup Engine. In FIGURE 8, the Ingress AGE and Egress AGE shares a common Bi-directional Lookup Engine and a common Flow Database.

**[0063]** FIGURE 9 illustrates the processing flow of the client frame through the AGE subsystem blocks. First, a received packet is processed by the Header Unit. The Header Unit extracts the two-tuple search key from the client frame according to the frame format configuration of the input channel. The search key is then passed on to the Lookup Engine. The Lookup Engine performs a wildcard linear search of the key in the

flow database. The output of the Lookup Engine is a multi-field flow context. The flow context is then used by the Tag Editor to perform tag modification on the original client frame. The modified frame is then buffered in the Flow FIFO according to the flow context. The scheduler of the Flow FIFO decides when to transmit packets into the output channels from which FIFO according to the output channel status and the Flow QoS parameters.

**[0064]** The Ingress/Egress Header Units will now be described. The AGE can support a number of different tag formats for flow identification. Different types of tags are supported at the Ethernet and the SONET/SDH side as shown in TABLE 1 and TABLE 2 below. TABLE 1 details tag format on the Ethernet interface. The extracted flow tag is combined with the Ethernet Channel (Port ID), or the SONET/SDH channel (or VCG) ID. It should be understood that while SONET VCG is discussed herein, all suitable SONET structures including, but not limited to, synchronous transport signals (e.g., STS-3) channels STS may be used within the scope of the present invention.

The term "encapsulation scheme" is interchangeably referred to as a "format" (see e.g. para. [0059], and a "protocol" (see e.g. paras. [0011], [0012] and [0013]. Applicant submits that anyone of skill in the art would reasonably understand these terms to be synonymous with the claimed "encapsulation scheme".

#### **Claim 4**

Amended claim 4 reads:

4. (previously presented) An Aggregation/Grooming Engine (AGE) for use within an Ethernet over SONET/SDH system (EOS system), said AGE comprising:
  - an ingress portion having
    - an ingress header unit for receiving data from an Ethernet MAC subsystem and extracting 2-tuple ingress search keys including a port or channel ID and an ingress frame tag, wherein

said ingress frame tag is according to an ingress frame tag encapsulation scheme;

an ingress lookup engine including a corresponding ingress flow database and coupled to said ingress header unit;

an ingress tag editor coupled to said ingress lookup engine;

and

an ingress flow FIFO unit coupled to said ingress tag editor

and an encapsulation engine; and

an egress portion having

an egress header unit for receiving data from said

encapsulation engine and extracting 2-tuple ingress search keys including a virtual concatenation group ID and an egress frame tag, wherein said egress frame tag is according to an egress frame tag encapsulation scheme;

an egress lookup engine including a corresponding egress flow database and coupled to said egress header unit;

an egress tag editor coupled to said egress lookup engine;

and

an egress flow FIFO unit coupled to said egress tag editor

and said Ethernet MAC subsystem;

wherein said ingress portion and said egress portion of said AGE provide grooming and aggregation functionality for said EOS system including label lookup, flow buffering, label editing, and flow scheduling.

Support for the amendments to claim 4 can be found at least at paras. [0069] - [0072], which describe in detail the extraction of two-tuple search keys:

**[0069]** The extracted flow tag is combined with the Ethernet Channel (Port) ID, or the SONET/SDH VCG channel ID to form a search key of two-tuple < CID, FlowTag>. For the Ingress Header Unit, the CID field represents from which Ethernet port the packet is received, the FlowTag represents the flow tag extracted from the received packet. For the Egress Header Unit, the CID represents from which Virtual Concatenation Group the packet is received, and the FlowTag represents the flow tag

extracted from the received packet according to the channel tag format configuration.

**[0070]** The Ingress/Egress Lookup Engine will now be described involving one embodiment as a Unidirectional Lookup Engine. In such embodiment of the invention, the ASE uses separate Ingress and Egress Lookup Engine as shown in FIGURE 7. The Ingress Lookup Engine and Egress Lookup Engine are two variations of the same design. The lookup procedures and the format of flow databases are the same between the Ingress Lookup Engine and Egress Lookup Engine, but the source of the lookup key and the content of the flow databases are different.

**[0071]** As an example, the ASE Ingress lookup database shown in FIGURE 10 is arranged to represent the flow definitions shown in FIGURE 6. The database is a two dimensional linear table. Each row of the table represents the definition of a client flow. Each column of the table represents a parameter field associated with the flow. The parameter fields are:

- a. iCID: Input Channel ID, represents which input channel the frame is received from. It is used to match against the iCID field of the two-tuple search key.
- b. iFlowTag: Input Flow Tag, represents the tag value that identifies the client flow. It is used to match against the iFlowTag field of the two-tuple search key. The iFlowTag can be either an exact value, a wildcard value that represents a range.
- c. oCID: Output Channel ID, represents which output channel the flow should be sent towards.
- d. oFlowTag: Output Flow Tag, represents the tag value to be added to the outgoing frame, to be used as an operand in the packet editing operation.
- e. TagCmd: Tag Editing Command, represents the tag operation to be done on the frame. Possible values are: POP (remove iFlowTag), NOP (keep Tag), PUSH (Add oFlowTag), REPLACE (iFlowTag with oFlowTag).

**[0072]** The <iCID, iFlowID> belongs to the Flow Key Fields of the database. The CTagCmd, oCID, oFlowTag, FlowID> constitutes the Flow Context Fields determines the operations on the client frame in the Tag Editor and FlowFIFO subsystems of the AGE.

### **Claim 9**

Amended claim 9 reads:

9. A method of packet grooming and aggregation within an Ethernet over SONET/SDH system (EOS system), said method comprising:

- receiving a data packet tagged according to an encapsulation scheme and including a port or channel ID;
- providing an input client frame from said data packet to a header unit;
- extracting a search key including said port or channel ID and said tag from said input client frame via said header unit;
- correlating said search key via a lookup engine to a match in a flow database to determine flow context;
- modifying said input frame via a tag editor according to said flow context;
- buffering said input client frame via a flow FIFO;
- applying discard policies to said flow FIFO based on said flow context; and
- scheduling said input client frame via a scheduler of the flow FIFO for transmission into output channels according to output channel status and flow quality of service parameters.

Support for this amendment can be found at least at paras. [0055], [0064], [0069] and [0084] and in Figure 13, which are reproduced below:

**[0055]** The Encapsulation Engine deals with the adaptation between Ethernet frames and SONET/SDH byte stream using adaptation protocols such as GFP, X.86, or PPP. At the ingress direction (from Ethernet to SONET/SDH), the Encapsulation Engine takes variable length Ethernet

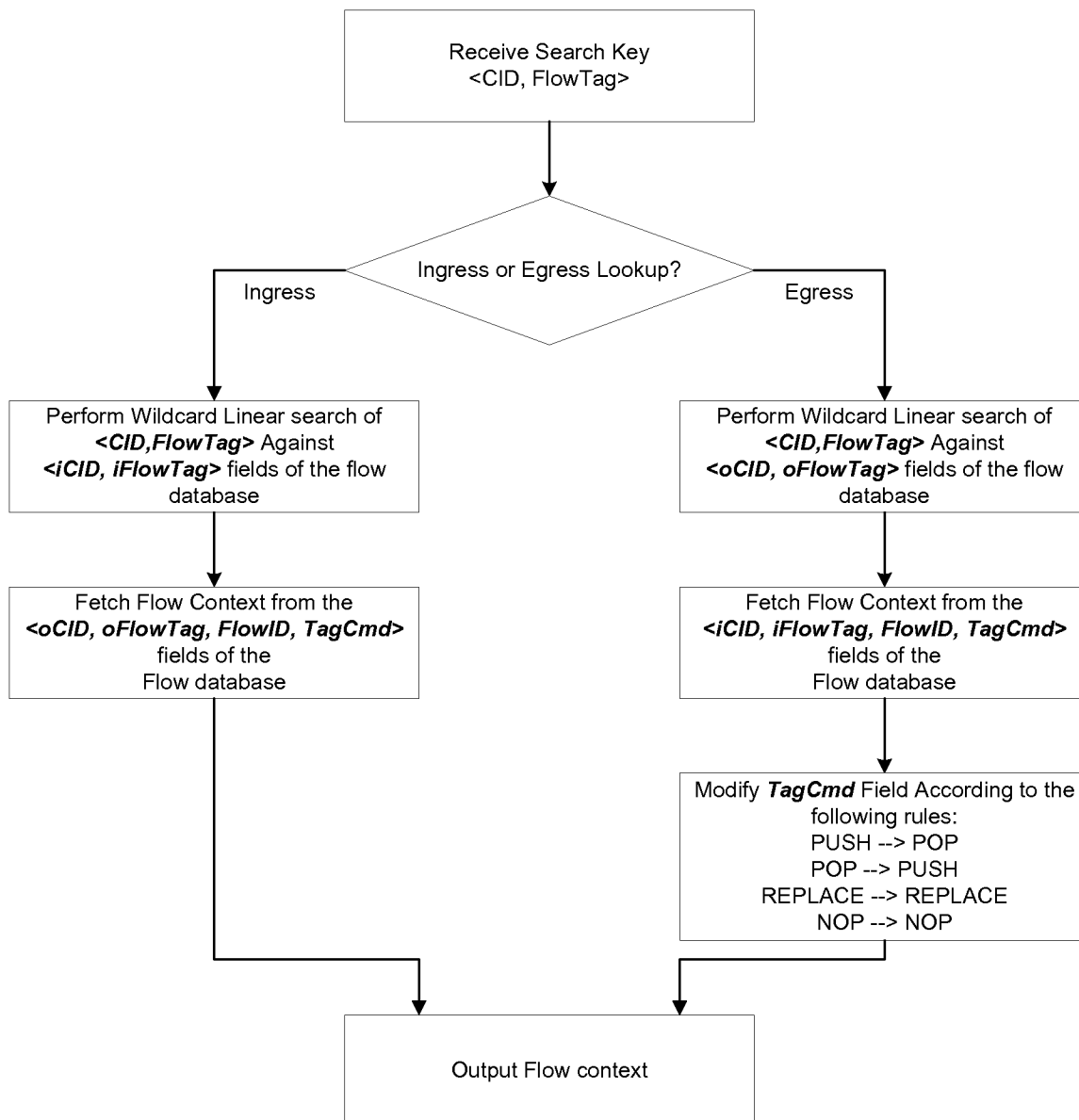
frames, and encapsulates the frame into GFP, X.86, PPP, or HDLC framing format. By adding the HEC (header checksum) or HDLC escape sequence, the framing structure allows the remote end to delineate the frames when the byte stream is received. At the egress direction (from SONET/SDH to Ethernet), the Encapsulation Engine takes the byte streams received from the SONET/SDH transport pipe and recovers the Ethernet frames using the defined framing procedures. The Encapsulation Engine also deals with the control signalling protocols for set-up, tear-down, and maintenance of the connection and for communication of control information between the two ends of the connection.

**[0064]** The Ingress/Egress Header Units will now be described. The AGE can support a number of different tag formats for flow identification. Different types of tags are supported at the Ethernet and the SONET/SDH side as shown in TABLE 1 and TABLE 2 below. TABLE 1 details tag format on the Ethernet interface. The extracted flow tag is combined with the Ethernet Channel (Port ID), or the SONET/SDH channel (or VCG) ID. It should be understood that while SONET VCG is discussed herein, all suitable SONET structures including, but not limited to, synchronous transport signals (e.g., STS-3) channels STS may be used within the scope of the present invention.

**[0069]** The extracted flow tag is combined with the Ethernet Channel (Port) ID, or the SONET/SDH VCG channel ID to form a search key of two-tuple < CID, FlowTag>. For the Ingress Header Unit, the CID field represents from which Ethernet port the packet is received, the FlowTag represents the flow tag extracted from the received packet. For the Egress Header Unit, the CID represents from which Virtual Concatenation Group the packet is received, and the FlowTag represents the flow tag extracted from the received packet according to the channel tag format configuration.

**[0084]** The Ingress/Egress FIFOs implement the client flow frame buffering and scheduling function. The function of the Ingress Flow FIFO and Egress Flow FIFO is essentially the same except they service the

traffic on opposite directions. The Flow FIFO is a multi-channel FIFO frame buffer. Each channel FIFO is logically a First-In-First-Out memory. Each client flow consumes one dedicated channel of the FIFO. At the input of the FIFO, the client frames are sent to the FIFO channel identified by FlowID in the flow context generated by the AGE Lookup Engine. The frame is stored in the FIFO memory according to the queuing discipline imposed on the FIFO.



**FIGURE 13**

**Claim 10**

Amended claim 10 reads:

10. The method of packet grooming and aggregation as claimed in claim 9 wherein said scheduling step occurs in accordance with said flow context.

Support for the amendment to claim 10 can be found at least at para. [0084] of the specification as filed:

**[0084]** The Ingress/Egress FIFOs implement the client flow frame buffering and scheduling function. The function of the Ingress Flow FIFO and Egress Flow FIFO is essentially the same except they service the traffic on opposite directions. The Flow FIFO is a multi-channel FIFO frame buffer. Each channel FIFO is logically a First-In-First-Out memory. Each client flow consumes one dedicated channel of the FIFO. At the input of the FIFO, the client frames are sent to the FIFO channel identified by FlowID in the flow context generated by the AGE Lookup Engine. The frame is stored in the FIFO memory according to the queuing discipline imposed on the FIFO.

#### ***Claim 14***

New claim 14 reads:

14. (new) The AGE as claimed in claim 4 wherein said ingress frame tag is an 802.1Q tag, a MPLS tag, or a proprietary tag.

Support for new claim 14 can be found at least at paras. [0064] - [0065] and Table 1 of the specification as filed:

**[0064]** The Ingress/Egress Header Units will now be described. The AGE can support a number of different tag formats for flow identification. Different types of tags are supported at the Ethernet and the SONET/SDH side as shown in TABLE 1 and TABLE 2 below. TABLE 1 details tag format on the Ethernet interface. The extracted flow tag is combined with the Ethernet Channel (Port ID), or the SONET/SDH channel (or VCG) ID. It should be understood that while SONET VCG is discussed herein, all suitable SONET structures including, but not limited to, synchronous transport signals (e.g., STS-3) channels STS may be used within the scope of the present invention.

**[0065] TABLE 1**

Flow Tag Format	Details
802.1 Q tag	Support both Ethernet V2.0 and Ethernet 802.3 frame format. The flow identifier can be the only 802.1 Q tag, or is the out-most tag in VLAN stacked frame with multiple 802.1Q tags. The inner 802.1Q tags are regarded part of the client payload.
MPLS tag	Support both Ethernet V2.0 and Ethernet 802.3 frame formats with or without 802.1Q tag.
Proprietary Tag	Support both Ethernet V2.0 and Ethernet 802.3 frame formats with or without 802.1Q tag. The Proprietary tag may be defined in fixed offset location of the frame payload.

**Claim 15**

New claim 15 reads:

15. (new) The AGE as claimed in claim 4 wherein said egress frame tag is an 802.1Q tag, a MPLS tag, a proprietary tag, or a GFP tag.

Support for new claim 15 can be found at least at paras. [0066] - [0068] and Table 2 of the specification as filed:

**[0066]** TABLE 2 details tag format on the Ethernet interface.

**[0067] TABLE 2**

Flow Tag Format	Details
802.1 Q tag	Support both Ethernet V2.0 and Ethernet 802.3 frame format. The flow identifier can be the only 802.1Q tag, or is the out-most tag in VLAN stacked frame with multiple 802.1Q tags. The inner 802.1Q tags are regarded part of the client payload.
MPLS tag	Support both Ethernet V2.0 and Ethernet 802.3 frame formats with or without 802.1 Q tag.

Proprietary Tag	Support both Ethernet V2.0 and Ethernet 802.3 frame formats with or without 802.1Q tag. The Proprietary tag may be defined in fixed offset location of the frame payload.  Can also be proprietary Tag in the HDLC
GFP Tag	Can be the GFP CID (Channel ID) or other Flow Ids defined in the GFP linear extension header.

**[0068]** The Ingress Header Unit and the Egress Header Unit parse the receive frames according to the frame format defined in TABLE I and TABLE 2 to locate and extract the flow tag from the frames. The detailed specification of the various frame formats and the offset location of the relevant tag is defined in the 802.1 Q, 802.3, GFP, and X.86 standards and are well understood by one skilled in the art. The Header Units follows the conventions defined in these standards for packet format parsing. In one embodiment of the present invention, the format recognized by each channel (Ethernet Port or SONET/VCG) can be configured separately. The header units employ different parsing state-machine for different ports accordingly.

Accordingly, Applicants submit that each of claims 1, 4, 9, 10, 14 and 15, and their respective dependent claims, is fully supported by the specification as initially filed, and request withdrawal of the rejections under 35 U.S.C. 112 first paragraph, and further request entry and reconsideration of the amendments submitted October 15, 2007.

No fee is believed due for this submission. However, Applicant authorizes the Commissioner to debit any required fee from Deposit Account No. 501593, in the name of Borden Ladner Gervais LLP. The Commissioner is further authorized to debit any additional amount required, and to credit any overpayment to the above-noted deposit account.

Respectfully submitted,

**LIAO, Heng et al.**

Borden Ladner Gervais LLP  
World Exchange Plaza  
100 Queen Street, Suite 1100  
Ottawa, ON K1P 1J9  
CANADA  
Tel: (613) 787-3519  
Fax: (613) 787-3558  
E-mail: [akinsman@blgcanada.com](mailto:akinsman@blgcanada.com)

By: /Anne Kinsman/  
**Anne Kinsman**  
**Reg. No. 45,291**

ALK/dbm